# A TEMPLATE SYSTEM PERSPECTIVE TO FASTER, LOWER COST AND QUALITY WEB APPLICATION DEVELOPMENT

Udai Arora
User Experience Studio, Hewlett Packard R&D,
Mahadevapura campus, Bangalore, India

***ABSTRACT***

*Web application development boils down to three major issues- Timelines, budget and the quality of the developed product. Minuscule and stringent deadlines and mostly limited budget have been an integral part of product development. Innovating products and improvising their quality has become a major challenge not only for small firms, but even for many larger establishments because of these constraints.*
*This research focuses on how product development can be accelerated by using template systems and thereby allowing more time for innovation to the organizations. Furthermore, comparison of a popular templating system- server side templating is made with a relatively new templating system- client side templating. These templating techniques were applied on a few use cases of the web-application developed and their performance based effect was studied with respect to the level of data interactivity in them. An attempt is made to draw a conclusion as to what templating system would be best suited for the development of web applications keeping in mind the limited budget and at the same time maximizing the performance of the web application and dealing with premature scalability issues.*

***KEYWORDS:*** *Web Application, Templating, Client Side Template System, Server Side Template System*

## I. INTRODUCTION

A web application is an application that is accessed over a network such as the Internet or an intranet. The term may also mean a computer software application that is coded in a browser-supported language (such as JavaScript, combined with a browser-rendered mark-up language like HTML) and reliant on a common web browser to render the application executable [1].
As web based-applications become increasingly important to all aspects of life, there is a need to encourage practitioners to adopt best practices so as to improve the quality of the processes in use, and therefore achieve targets related to time, budget and quality.
Most of web application development methodologies used these days are the extensions of standard software engineering methodologies. The usual waterfall model is too rigid an approach for developing web applications. The waterfall model process was perfect for developing a file maintenance program for mainframes, but far too restrictive a process for building a Web application. Web application development needs to be an iterative and progressive process.
The Agile method of development is becoming increasingly popular with software firms as it decreases the development time and promotes evolutionary development besides promising a rapid and flexible response to change. It diminishes the overall risk as a working product is available at the end of each iteration. New "features" are added with every iteration. Templating systems can be used to add the new features because of the numerous advantages they offer.
Section II attempts to explain templating systems and provides the working of the two major templating systems-The server side template system and the client side template system. Section III benchmarks these templating systems with respect to various levels of interactivity in web applications. Conclusion is discussed in section IV followed by future work in section V.

## II. TEMPLATING SYSTEMS

A web template system is used to process the web templates and produce web pages for deployment. At the heart of the web template system lays the template engine, which does the actual rendering of the web pages from the templates. The web page can be divided into many parts or templates. In other words, a combination of templates makes up a web page.

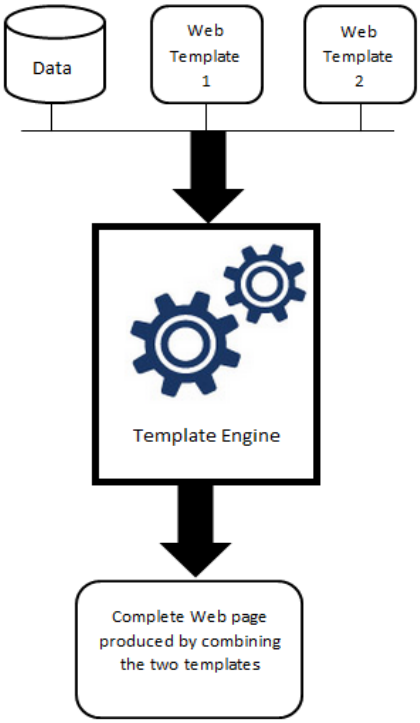


**Figure 1.** Template System

A web template system describes the methodologies used to produce web-pages with the help of a templating engine. The two major template systems are the server side and the client side systems. In the server side template model, the templates are rendered at the server itself and the mark-up generated is forwarded to client as opposed to the client side model where the mark-up is generated at the client side itself. The following diagram shows the working of the two template systems.

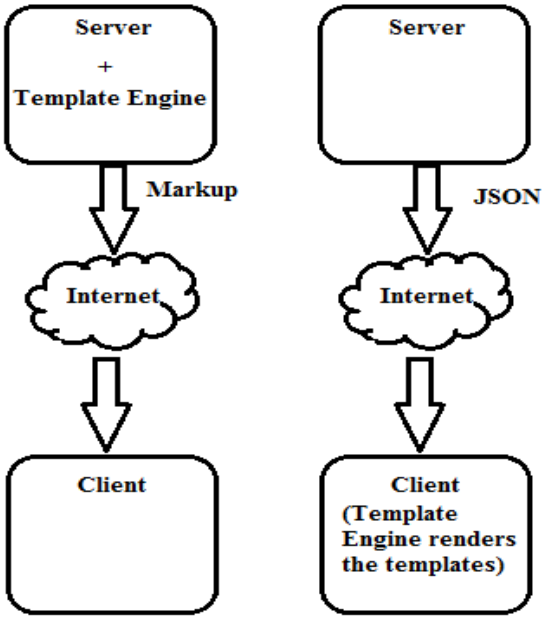


**Figure 2.** A Server side (left) and a Client Side Template System

A survey on web based project development by the Cutter Consortium highlighted problems for Web-based projects, which included the schedule delays which plagued the projects 79 percent of the time. Also, the deliverables were of poor quality 52 percent of time [2][3]. When we use templating system in agile development, we get a highly flexible model in which changes can be done as and when required and more templates can be added with each iteration. The different templates can be worked on in parallel by the developers. Since templates abstract the business logic from the presentation, you can also delegate the responsibilities of the presentation to designers. The designers can create and modify templates, and the developers can work in the logic almost simultaneously, resulting in much faster development and hence considerably lower development time. It is also easier to keep the presentation consistent.

## III. BENCHMARKING - RESULTS AND DISCUSSIONS

Consider 4 web applications- one extremely interactive with data, another a little less data interactive, one barely interactive and one completely static. The observations on the following factors are presented for each of these web applications:

### 3.1. Performance

If you are making an interractive application in which you expect the JS code to interract with the data, which is the case with most of the web applications, full client rendering would be ideal as it minimises the number of request to the server resulting in increased performance. In client-side template systems, you just need the raw data that has changed. In server-side template systems, you are forced to send the entire section even if it might just be a single element that has changed.
Eg- Consider a messaging app. If the whole app is rendered on the server, clicking on a tab in the app requires a lot of re-rendering- the app has to hit the server, render the HTML and return it. Now if the same app uses client side rendering, rather than getting rendered HTML from the server, it just gets the JSON for the chat, which is considerabily smaller in size, and renders it with Javascript at the client, therby saving a lot of bandwidth for the user resulting in increased performance.
The following graphs illustrate for the different levels of data interactivity, the total relative time to transfer + render + display the web application for first few requests for both- server side and client side template systems. In each case, the 3rd request for the cleint side template system is normalised to unity and all other requests are relative to this request.

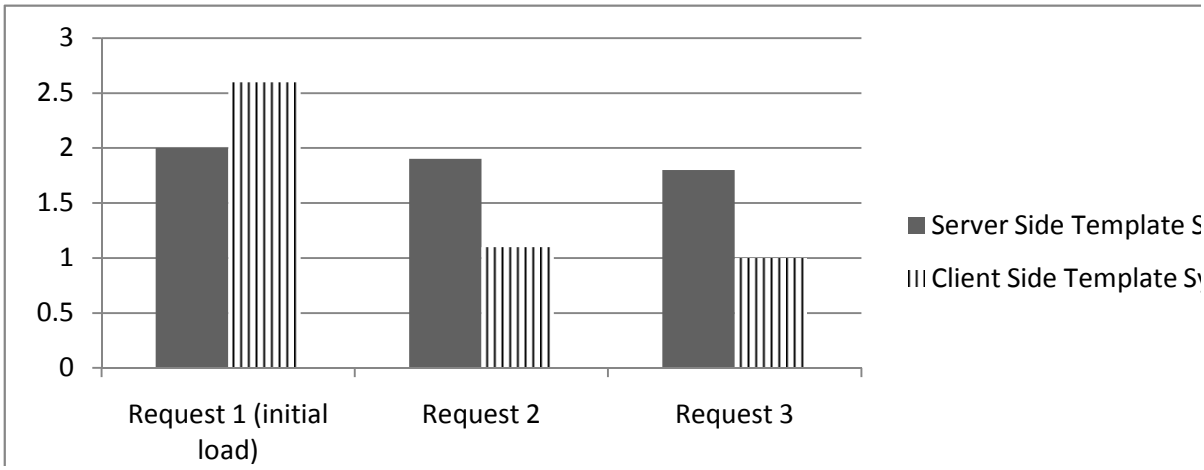


**Figure 3.** Highly data interactive web app

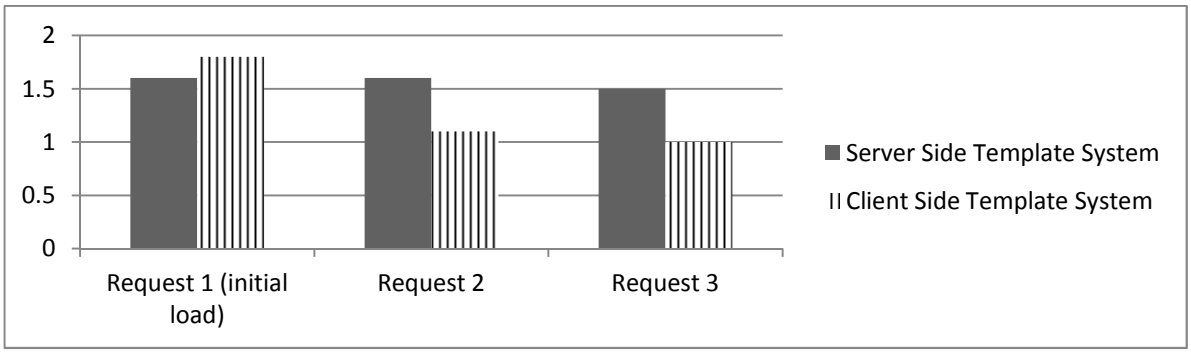


**Figure 4.** Moderately data interactive web app

However, if you have more static content and an efficient server, a server side template system will be more advantageous as it won't rely on client for rendering and transferring static content will take almost the same bandwidth with mark-up as with JSON.

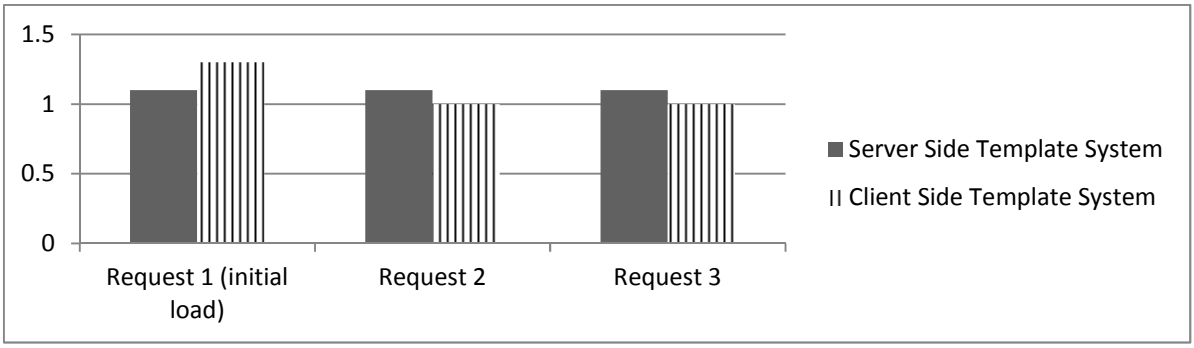


**Figure 5.** Barely data interactive web app

For a completely static page, all the requests will essentially require the same time in both the template systems.

### 3.2. Scalability

Since the number of request to the server are minimised in the client side system, the same server is able to handle more users, making it somewhat more scalable. However if the web application is essentially static, the number of requests to the web server will be almost the same in both the template systems, scaling equally in both the cases.

### 3.3. Abstraction and maintenece:

The Javascript API separates concerns: HTML, CSS and Javascript are logically separated in their own files. This keeps mark-up and styling out of the Javascript logic, so adding functionality to the User Interface does not require adjusting the mark-up or the styles, and vice-versa. As a result, maintenance is easier in case of the client side template system.

## IV. CONCLUSIONS

Web Application Development using template systems takes place at a much faster rate, which can help meet the rigorous deadlines. Except for the initial loading of the page, the client side template systems prove to be much faster than its server side counterpart and this difference is directly proportional to the amount of data interactivity -for small to medium sized data. Furthermore, if we use client side template systems, we can not only save on the additional servers required by the server side system, but also on the costly server side caching. Hence, the client side template system proves to be more beneficial than its server side counter-part in most of the cases.

However, the developers should use a fall-back approach if the time and budget constraints are not too stringent. In the fall-back approach, if web application uses client side template system, they should make sure that it safely fall-backs to a server side setup in case the client cannot render at its end due to disabled Javascript or any other reason.

## V. FUTURE WORK

Research on hybrid template systems can be carried out to determine a sweet spot- the perfect balance of server and client side template systems with respect to data interactivity level of applications, so as to further improve the performance related aspects.

## ACKNOWLEDGEMENTS

I would like to thank Amit Pande, Uday MS, Devanand from Hewlett Packard for their guidance through this research.

## REFERENCES

[1]. Daniel Nations, "What is a web application"
[2]. Beck, Kent, (2001) "Manifesto for Agile Software Development", Agile Alliance
[3]. Parr, Terence John, (2004) "Enforcing strict model-view separation in template engines" Proceedings of the 13th international conference on World Wide Web.
[4]. Haroon Altarawneh and Asim El Shiekh "A Theoretical Agile Process Framework for Web Applications Development in Small Software Firms" Sixth International Conference on Software Engineering Research, Management and Applications. p:125-131.
[5]. Abdesselam Redouane ," Guidelines for Improving the Development of Web-Based Applications", Proceedings of the Fourth International Workshop on Web Site Evolution (WSE'02) 0-7695-1804-4/02 2002 IEEE
[6]. Said Hadjerrouit, "Web-based Application Development: A Software Engineering Approach "ACM SIGCSE Bulletin June 2001 Vol 33. No. 2 p 31-34
[7]. Scott George M. and Zhiping Walter, "The Problems of Internet Systems Development", proceedings, Hawaii International Conference on Systems Sciences-35, 2002.
[8]. Pascal Deschenes, 2011, "Client Side Templating".
[9]. Abdesselam Redouane ,"Towards a New Method For The Development Of Web-Based Applications", Proceedings of the Third IEEE International Conference on Cognitive Informatics (ICCI'04) 0-7695-2190-8/04 © 2004 IEEE
[10]. Andrew McDonald and Ray Welland, 2002, " Evaluation of Commercial Web Engineering Processes"
[11]. Steven P. Dow, Julie Fortuna, Dan Schwartz, Beth Altringer, Daniel L. Schwartz, Scott R. Klemmer, "Prototyping Dynamics: Sharing Multiple Designs Improves Exploration, Group Rapport, and Results"

## AUTHOR

**Udai Arora** completed his graduation in Computer Science from Manipal Institute of Technology, Manipal. He is currently working in Hewlett Packard R&D, Bangalore. His research interest includes Internet Technologies, User Experience, Databases and Programming Languages.